High Integrity Software for High Integrity Systems

George Romanski
Romanski@Verocel.com
Outline

• System safety
• Safety standards
• Safety objectives
• Military avionics
• Ground based systems
• Future policy
• Directions
Brake-by-Wire Safety Solutions

- System Design - Limit speed to 5 mph
- Hardware Design - use mechanical/hydraulic backup
- Replicate Computer systems - no common mode errors
- Software assurance through compliance with standards (e.g. DO-178B)
DO-178B / ED-12B

• Acceptable means of compliance to the regulators of software in avionics systems

Not the only means of compliance!

But, if you choose a different approach

Must show DO-178B/ED-12B objectives have been met
Intent of DO-178B

• Describe objectives for Life-Cycle Processes
• Describe process activities
• Describe evidence required at different assurance levels
SC-190, WG-52 Committee

- 150 Registered Members
- Consensus based
- 4 years - Report published
  - Annual Report for Clarification of DO-178B (DO-248A)
  - Annual Report for Clarification of ED12B (ED-94A)
- Position Papers
  - Document corrections
  - FAQ’s (Clarifications)
  - Discussion Papers
- CNS/ATM - Work Continues
Typical DO-248A clarifications

• Is recursion permitted in airborne applications?
  – Yes, but it must be bounded (…etc)

• Is Source-code to Object-code traceability required?
  – Yes, if providing coverage analysis at source code and level A
  – No, if providing coverage at machine code

• If some run-time functions are inlined, is coverage still required?
  – Yes, cannot conceal coverage obligations
Typical DO-248A clarifications Cont.

• Can compiler features be used to simplify coverage analysis at object code?
  – Yes! (e.g. short-circuit operations)
  – But, the compiler (feature) is being used as a verification tool so compiler (feature) must be qualified as a verification tool

• What are the issues for reverification of COTS software?
  –
Standard Waterfall Process Model

Requirements → Design → Code

Where is the Evidence?

Test
Code Exists - Requirements re-engineered

1. Requirements
2. Design

1. Code
2. Test
Requirements Based tests

1. Requirements
2. Design
3. Code
4. Test
5. Develop Tests

Diagram:
- Requirements flows to Design
- Design flows to Code
- Code flows to Test
- Test flows to Develop Tests
Standard Waterfall Model

1. Requirements
2. Design
3. Develop Tests
4. Code

Materials Developed /Reviewed by Re-engineering
Validation and Verification

Validation

| Req 1 | Req 2 | Req 3 | Req 4 | Req 5 | Req 6 |

Verification

| Component 1 | Component 2 | Component 3 | Component 4 | Component 5 | Component 6 |

Goals

System built to Requirements

Complete and Correct
RTS an Important Component

System cannot be Certified unless RTS is Verified

SYSTEM in one address space

Application Programming Interface

Run-Time System

Application

Code

Same assurance level for all components
## Deterministic Behavior

<table>
<thead>
<tr>
<th>Functionality</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resources</td>
</tr>
<tr>
<td>Time</td>
</tr>
</tbody>
</table>
Deterministic Behavior

• Results of a function are the inevitable consequence of its inputs:
  – Parameters
  – Global variables

• Bound on the resources used
  – Memory - no new memory after startup
  – Stack - HUGE margins

• Bound on the time taken to complete the function
  – time taken to execute a function depends on many system level parameters,
  – non-linear relationships are noted as they can cause the application to miss deadlines
Black Box Testing

- No single failure should prevent “Continuous safe flight and landing.”

- Statistical testing cannot show absence of a single state that will cause a failure

- Software has discontinuities

- Software does not follow Gauss/Normal Distribution

There is no foundation for statistical reasoning about software faults or safety
Coverage Analysis

- Analysis of testing methods and results to show effectiveness of testing
- Method to show absence of unintended function
- Should be based (as much as possible) on requirements based tests
- Rigor depends on criticality level

Note: Coverage Analysis not Coverage Testing
Coverage at Level B and C

- Statement Coverage — Level C
- Decision Coverage
  - Entry Points
  - Exit Points
  - All Decisions
  - All Outcomes

Level B
Coverage at Level A

- Coverage required at Machine Code level or
- Show source to object code traceability and test at source level or
- Use different compilers and different languages or
- MCDC testing required
  - each condition must have effect on outcome
Military Avionics

- D0-178B - now mandated by congress
- Need Safety - even though:
  - Pilots have parachutes
  - Pilots don’t sue
- Want safe software
  - Don’t need the evidence ?
  - Must withstand an audit
The ‘Requirements’ for ATM Systems

<table>
<thead>
<tr>
<th>More</th>
<th>Safety</th>
</tr>
</thead>
<tbody>
<tr>
<td>Increase in</td>
<td>Capacity</td>
</tr>
<tr>
<td>Lower</td>
<td>Costs</td>
</tr>
<tr>
<td>Fewer</td>
<td>Resource constraints</td>
</tr>
</tbody>
</table>

Want to use COTS !!!
The ‘Challenges’ for ATM Systems

<table>
<thead>
<tr>
<th>Current technology</th>
<th>Becoming obsolete</th>
</tr>
</thead>
<tbody>
<tr>
<td>New Technology</td>
<td>Increasing in cost</td>
</tr>
<tr>
<td>Air Traffic in Europe</td>
<td>Increasing 6% pa.</td>
</tr>
<tr>
<td>Air Traffic in US</td>
<td>Increasing 4% pa.</td>
</tr>
</tbody>
</table>
WAAS

Ionospheric storm data

Selective availability helps

Sun may distort signal
The “Flight Profile”

Departure Procedure

Static Information
- Terrain
- Airways
- Airport

Dynamic Information
- Weather
- warnings
- capacity constraints
- Special use airspace schedules
- Etc.

Preferred Path

Preferred Climb

Preferred Descent

Airport
Object Oriented ‘Free-Flight’

- Flight Profile
- Filed Flight Trajectory
- Active Flight Trajectory
- Traffic Density Predictions
- Dynamic Route Structures
- Airspace Data
- Objects

VEROCHEL
Object Oriented Technology

• Pressure from industry to use it
• Industry expect lower certification costs - eventually
• Certification authorities nervous
Reusable Software Components (RSC)

- RSC Developer
- Integrator
  - Subsystem manufacturer

- RSC Run-Time system
- Product
  - e.g. FMS

- Applicant
  - Airframe manufacturer
  - Subsystem manufacturer

- FAA

- Product
  - e.g. Airplane, FMS
Reusable Software Component - Credit

- Applicant applies for Type Certificates for Product
- Applicant supplies DO-178B materials for RSC
  - Software Level (A, B, C, D)
  - Identified Processor type
  - Identified Compiler
- FAA provides letter to RSC developer which documents certification credit
- Eliminates / Reduces reverification on new project
Multiple Systems

- Cabin Management
- Power Management

1 box, 2 CPU’s

Primary ARINC Bus

Secondary ARINC Bus
Partitioned Systems

Primary ARINC Bus

- Cabin Management
- Power Management
- OS

Secondary ARINC Bus

- Cabin Management
- Power Management
- OS

Integrated Modular Avionics
APEX
ARINC 653

VEROCEL
High Integrity Software for High Integrity Software
The Partitioned Promise

- Cheaper to verify components
- Cheaper to re-verify components
- Lowers criticality level - lowers certification costs
- Less software to audit when component changed/upgraded
Don’t Argue with the Auditors

• Arguing with the auditors is like mud wrestling with a pig

• After a while you find out the pig really likes it!